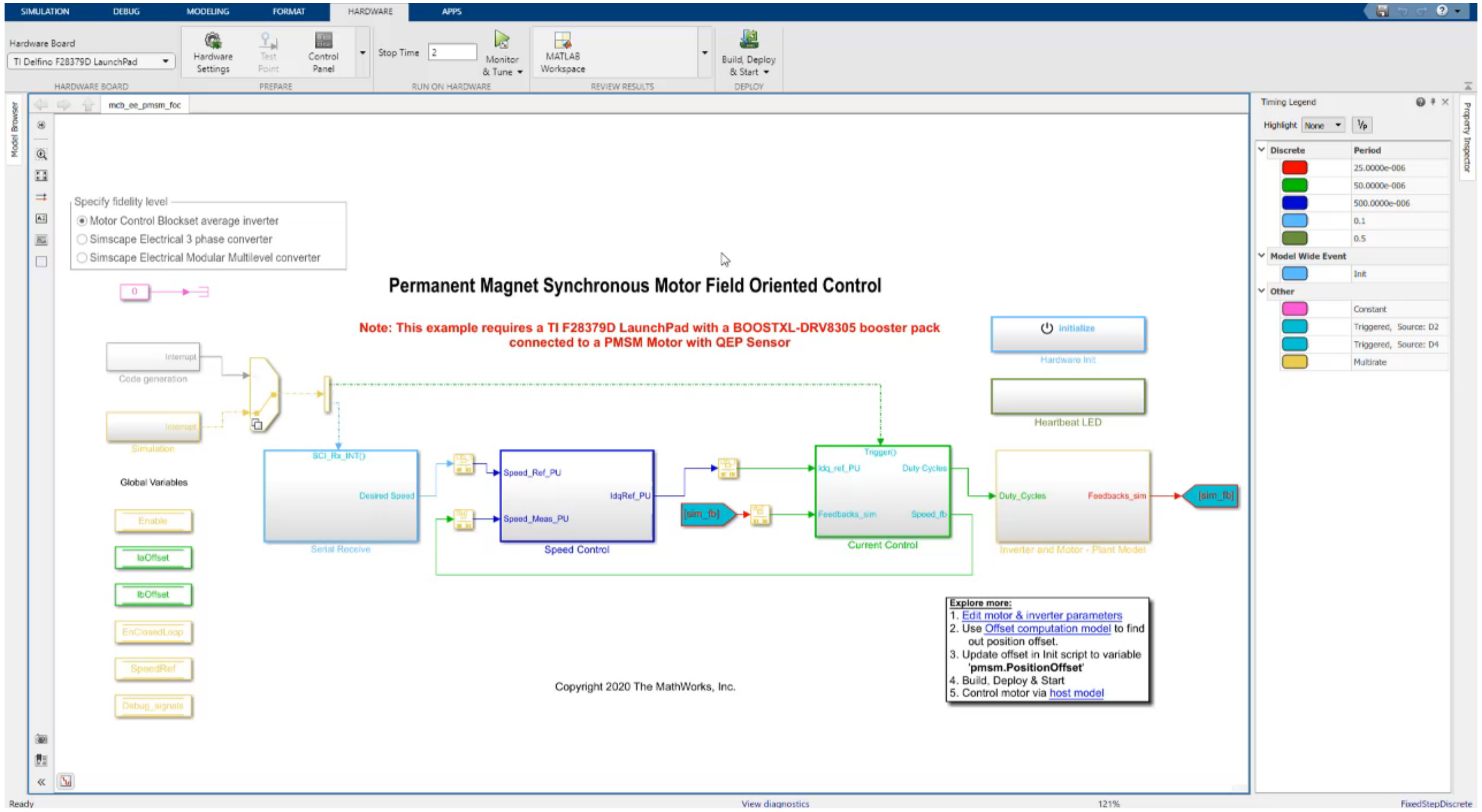
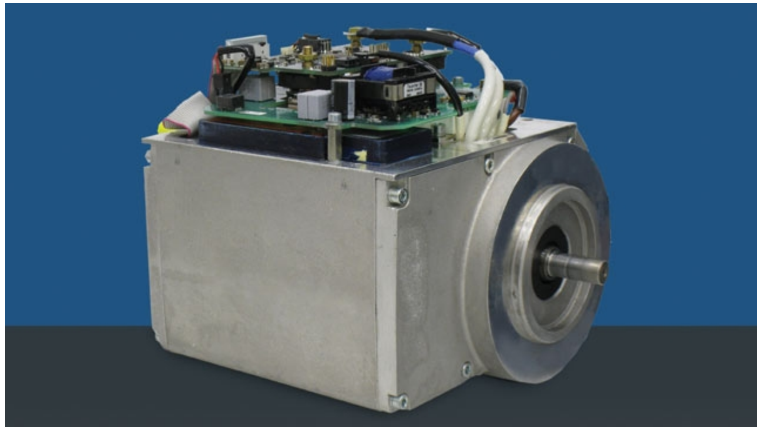
Deploying Motor Control Algorithms on NXP S32K from Simulink®

Brian McKay
Technical Marketing

We will spin a brushless motor using Simulink and Model-Based Design



Brushless motors are everywhere



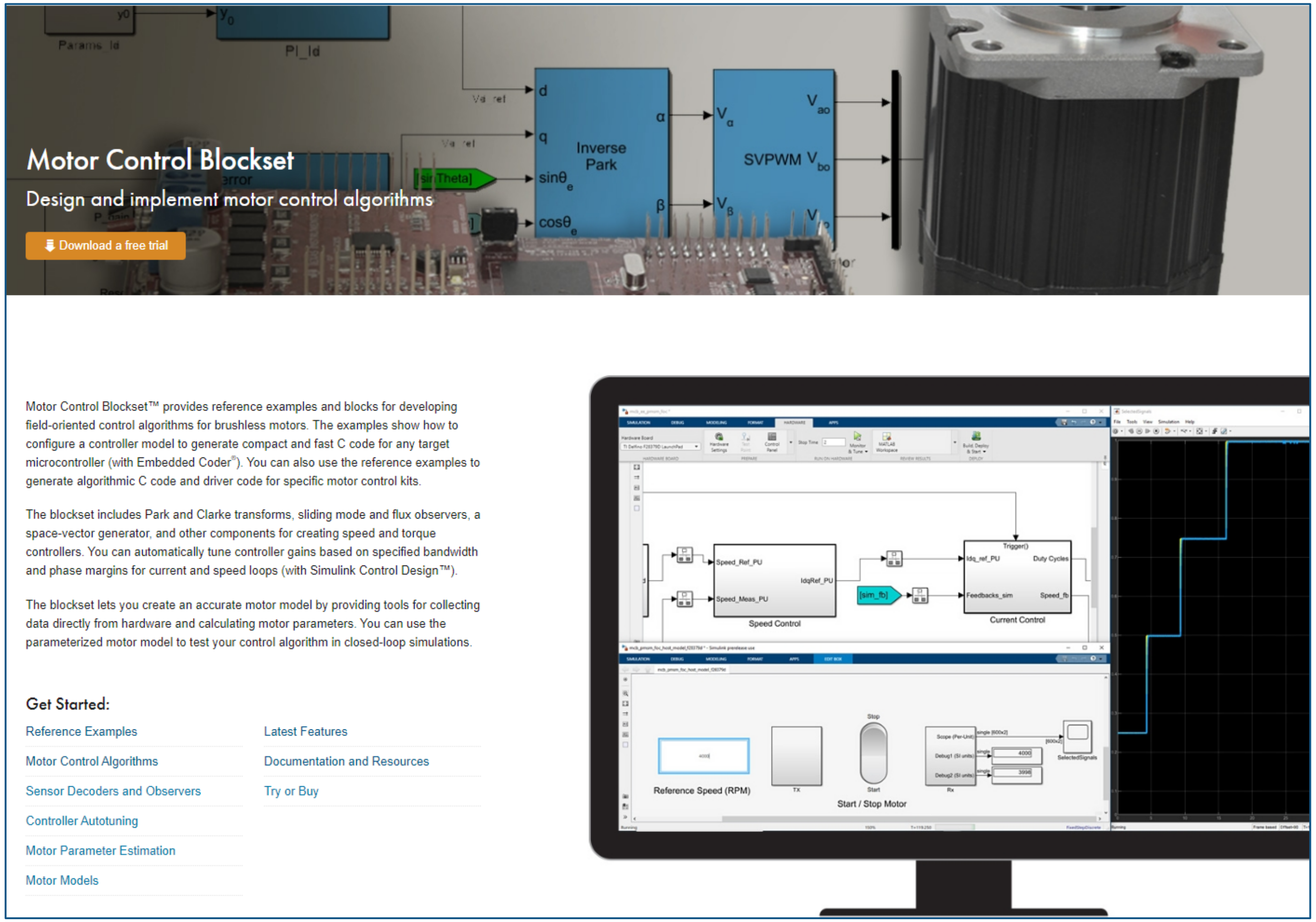
Why Simulink for motor control?

- Verify control algorithm with desktop simulation
- Generate compact and fast code from models
- Minimize development time using reference examples

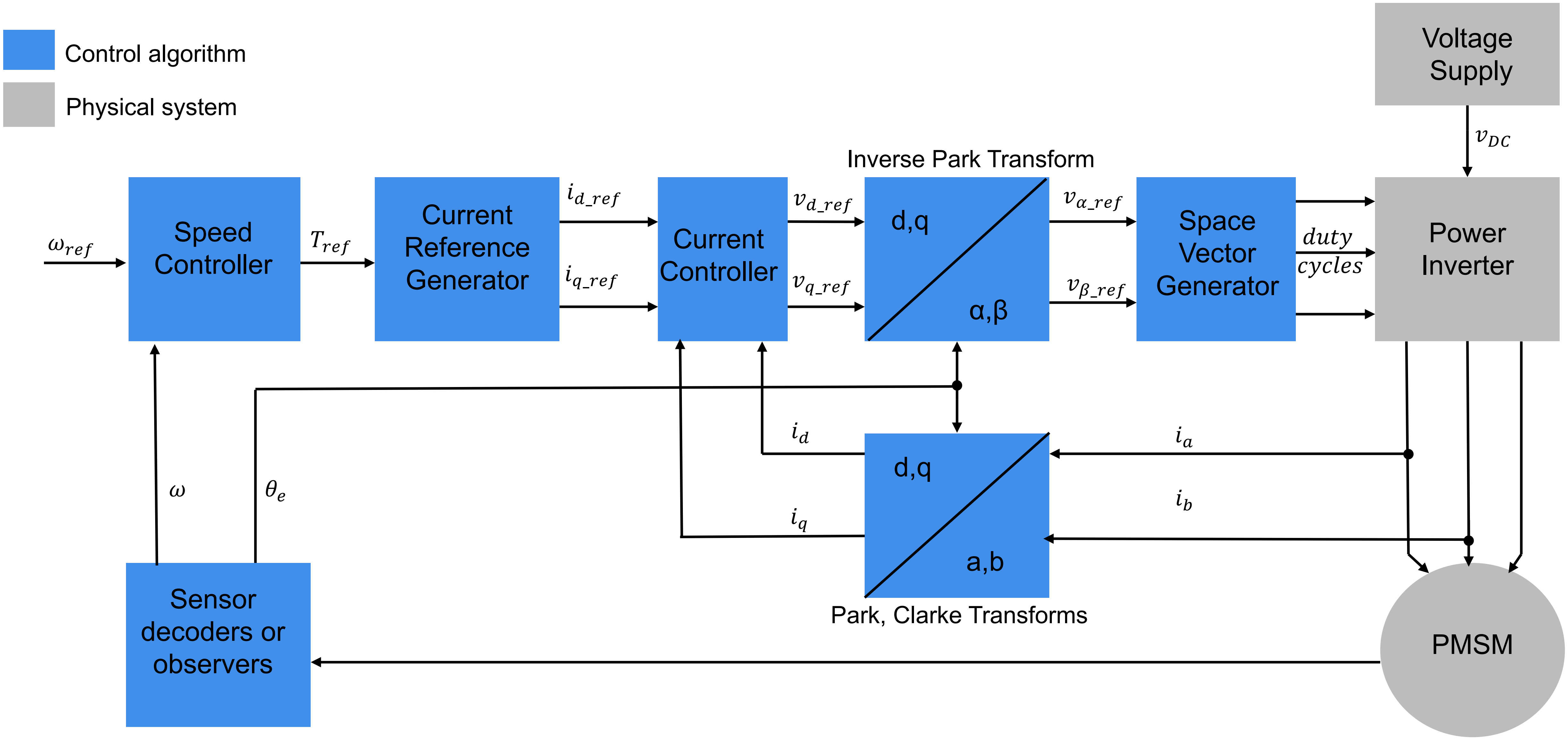
Customers routinely report 50% faster time to market

Motor Control Blockset simplifies the workflow

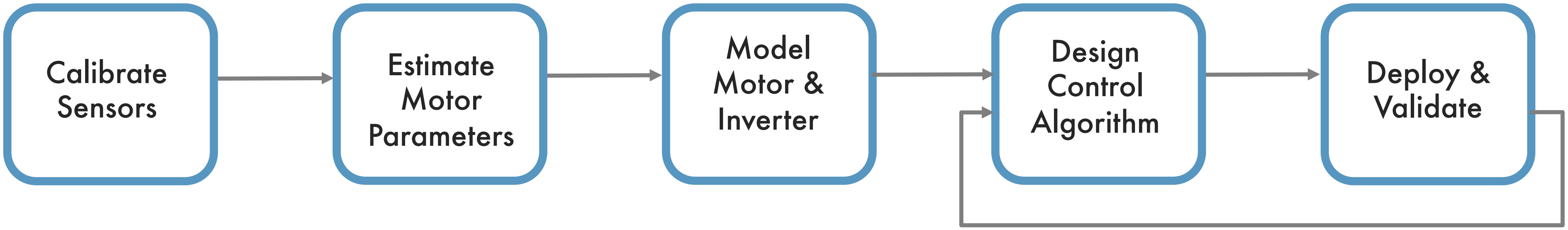
- Control blocks optimized for code generation
- Sensor decoders and observers
- Motor parameter estimation
- Controller autotuning
- Reference examples



Brushless motors require complex algorithms – field-oriented control



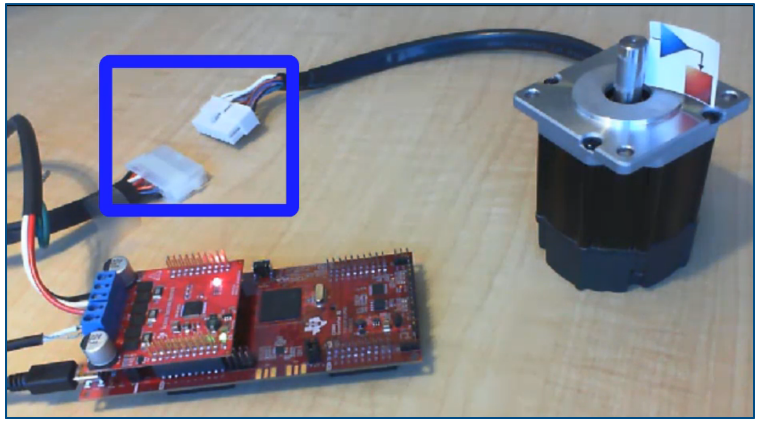
Workflow for implementing field-oriented control

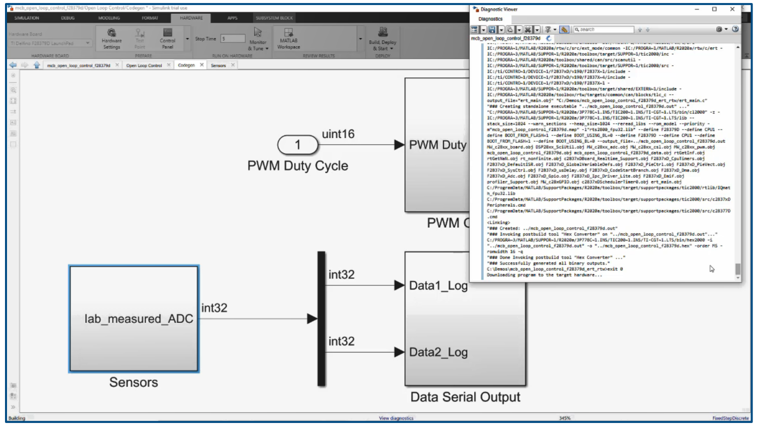


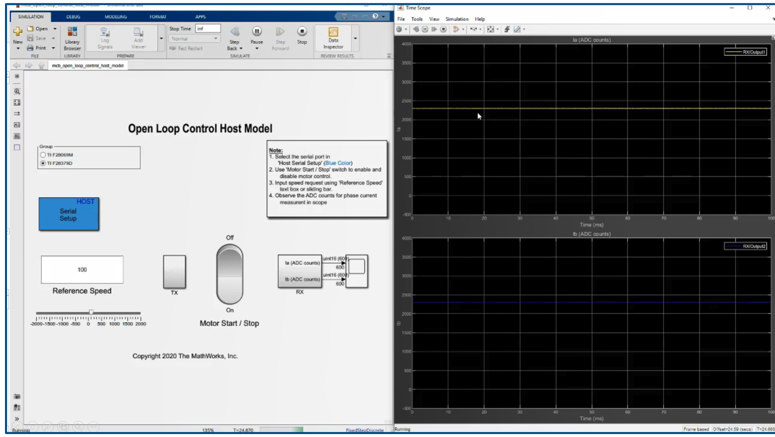
Sensor calibration

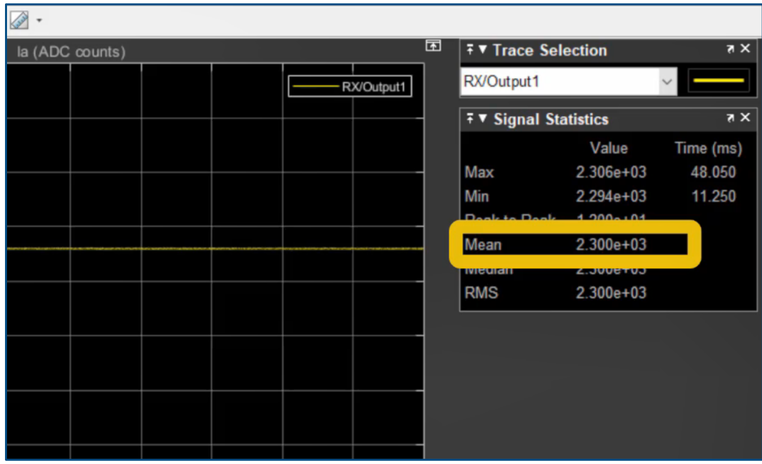
Calibrate Sensors → Estimate Motor Parameters → Model Motor & Inverter → Design Control Algorithm → Deploy & Validate

- Calibrate ADC offsets



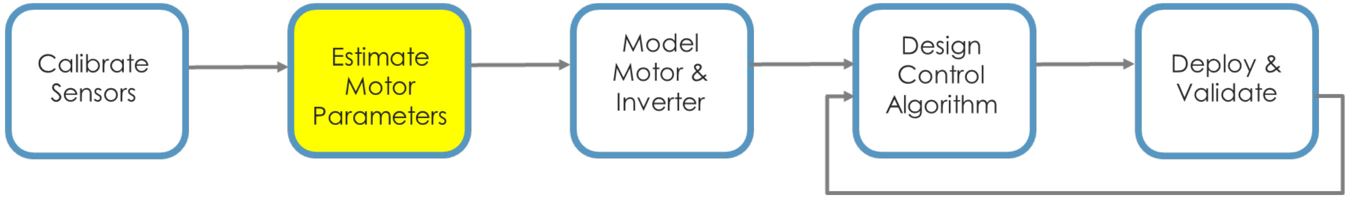




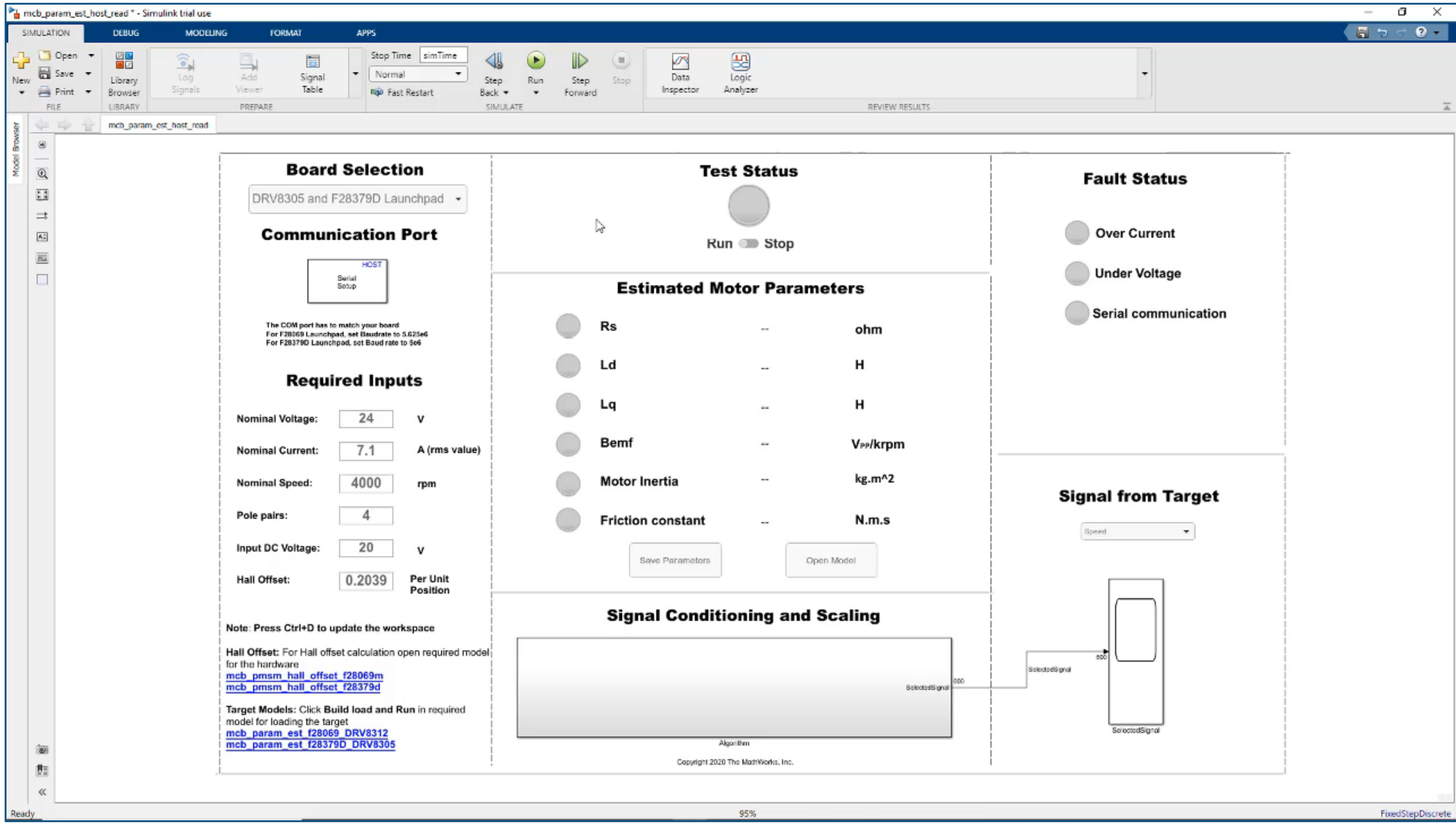



```
case 'BoostXL-DRV8305'
    inverter.model          = 'BoostXL-DRV8305';% M
    inverter.sn             = 'INV_XXXX';          %
    inverter.V_dc           = 24;        %V         //
    inverter.I_max          = 19.3;      %Amps      //
    inverter.I_trip         = 10;        %Amps      //
    inverter.Rds_on         = 2e-3;      %Ohms      //
    inverter.Rshunt         = 0.007;     %Ohms      //
    inverter.MaxADCCnt      = 4095;      %Counts    //
    inverter.CtSensAOffset  = 2300;         %Counts
    inverter.CtSensBOffset  = 2303;         %Counts
    inverter.ADCGain        = 1;         %          //
```

Parameter estimation

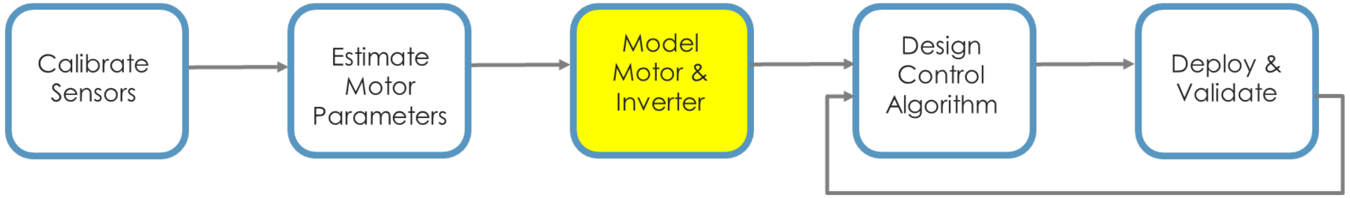


- Instrumented tests running on the target
- Host model to start and control parameter estimation



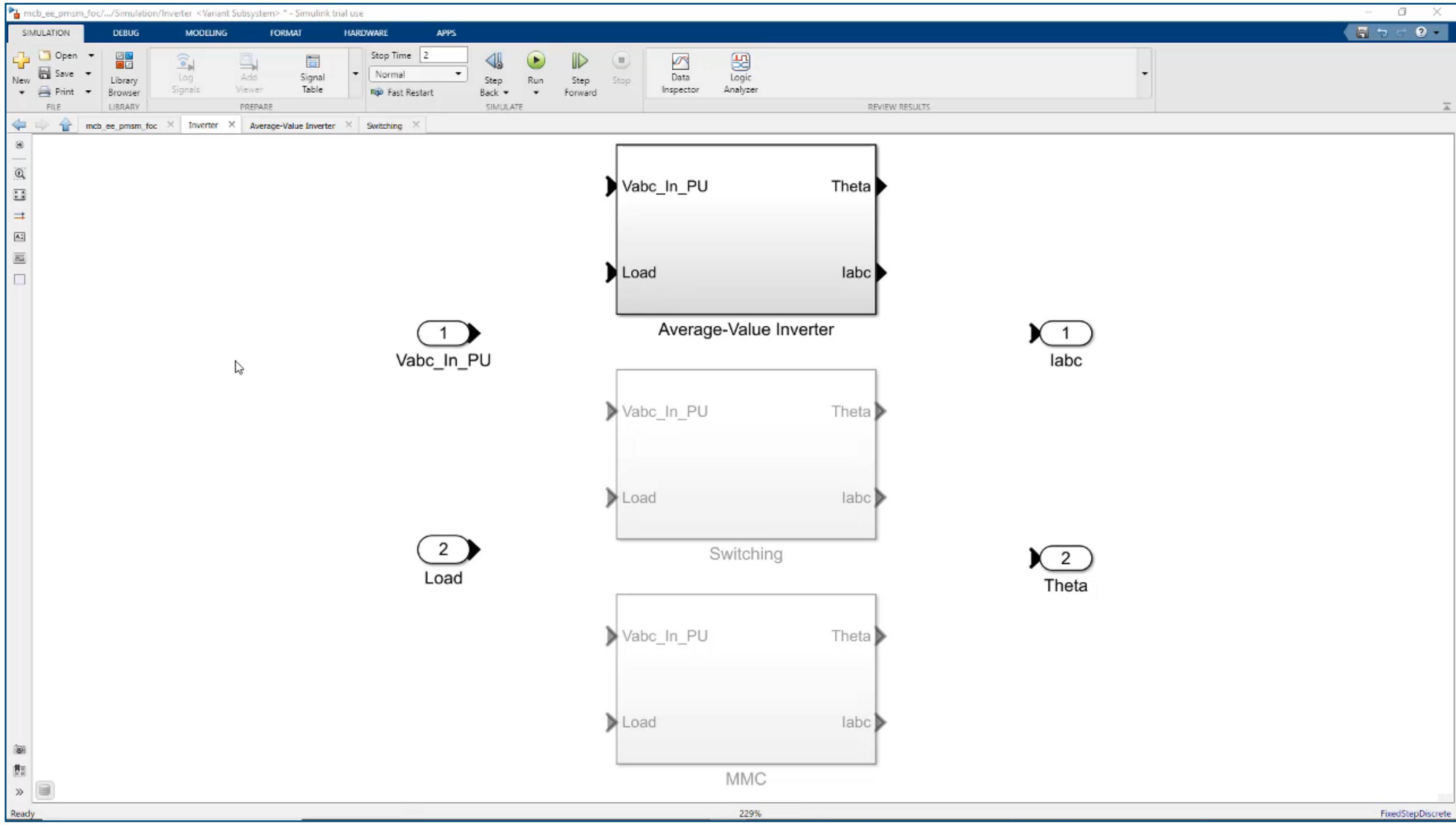
** Parameter estimation is supported by quadrature encoder, Hall sensors, or sensorless flux observer*

Modeling motor and inverter

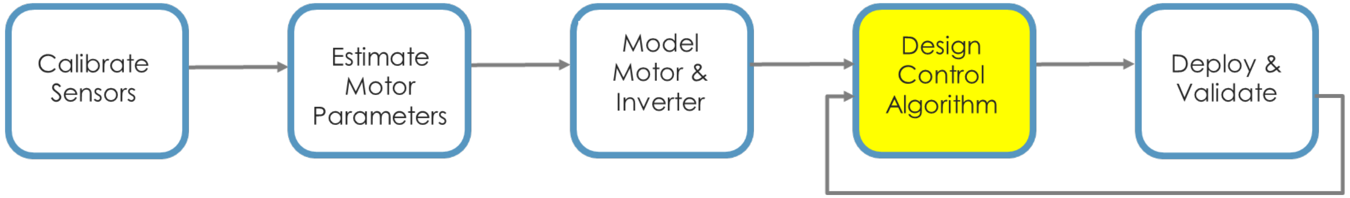


- Use linear lumped-parameter motor model

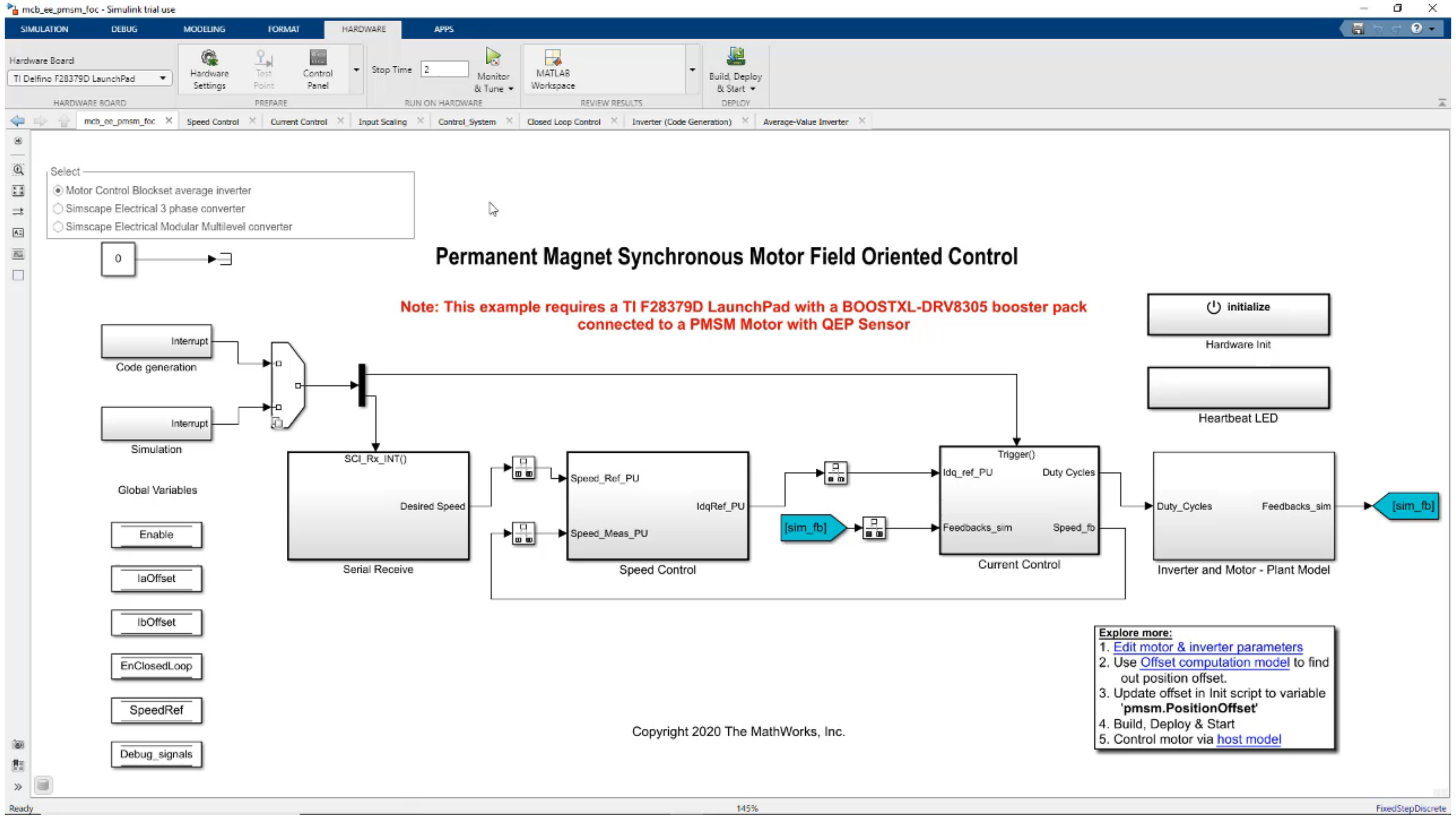
- Model inverter as an average-value inverter or model switching with Simscape Electrical



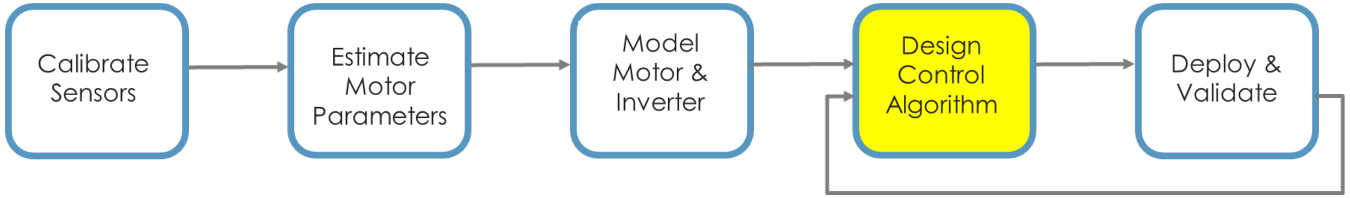
Control algorithm design



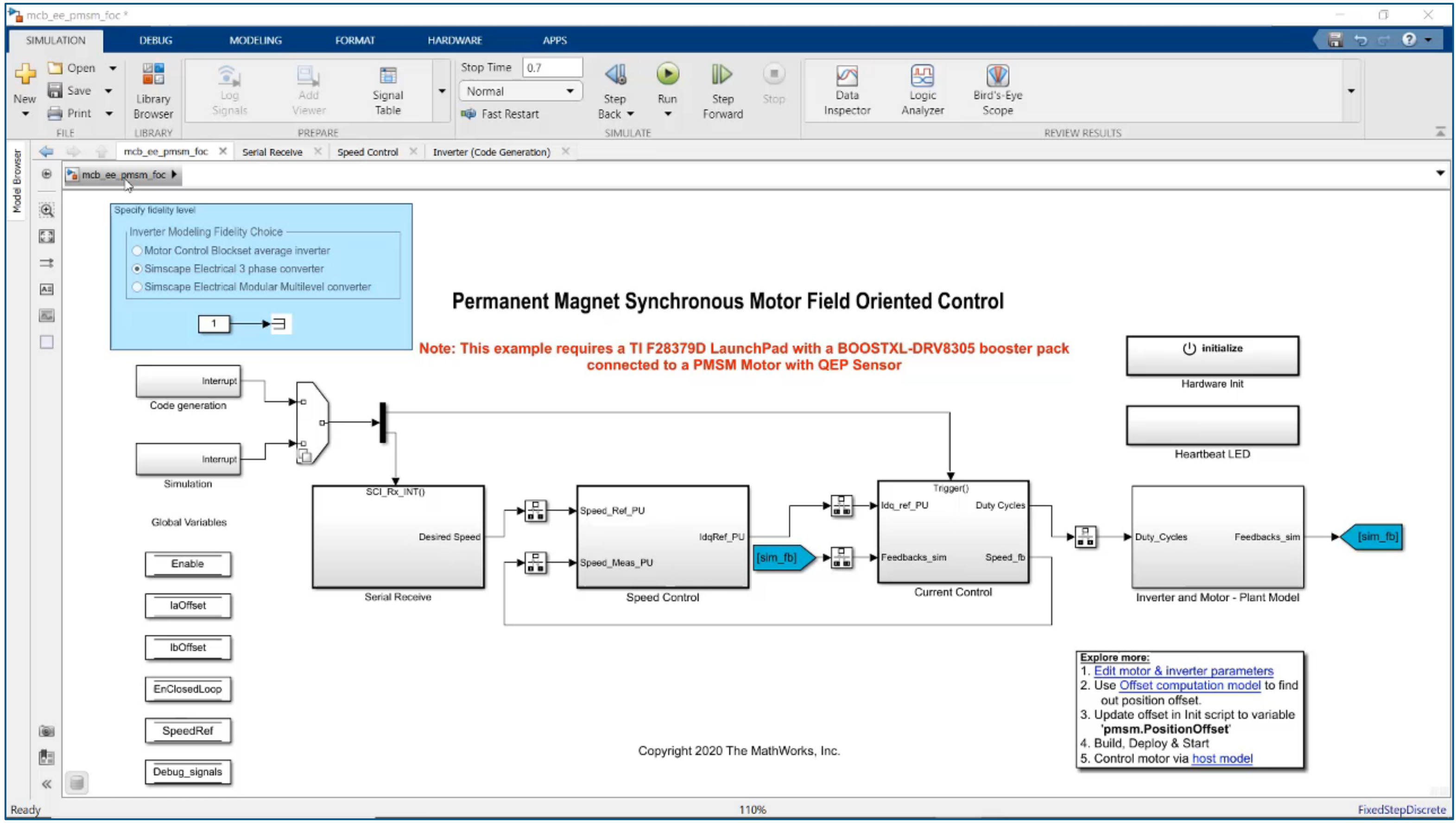
- Model field-oriented control algorithm
- Model sensor decoders or sensorless observers
- Tune loop gains
- Verify in closed-loop simulation



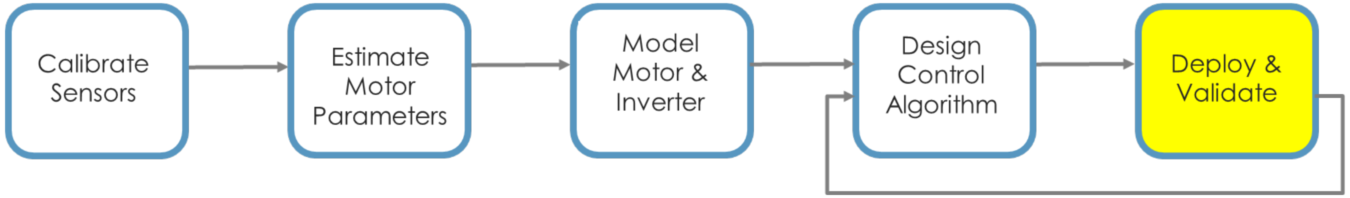
Control algorithm design



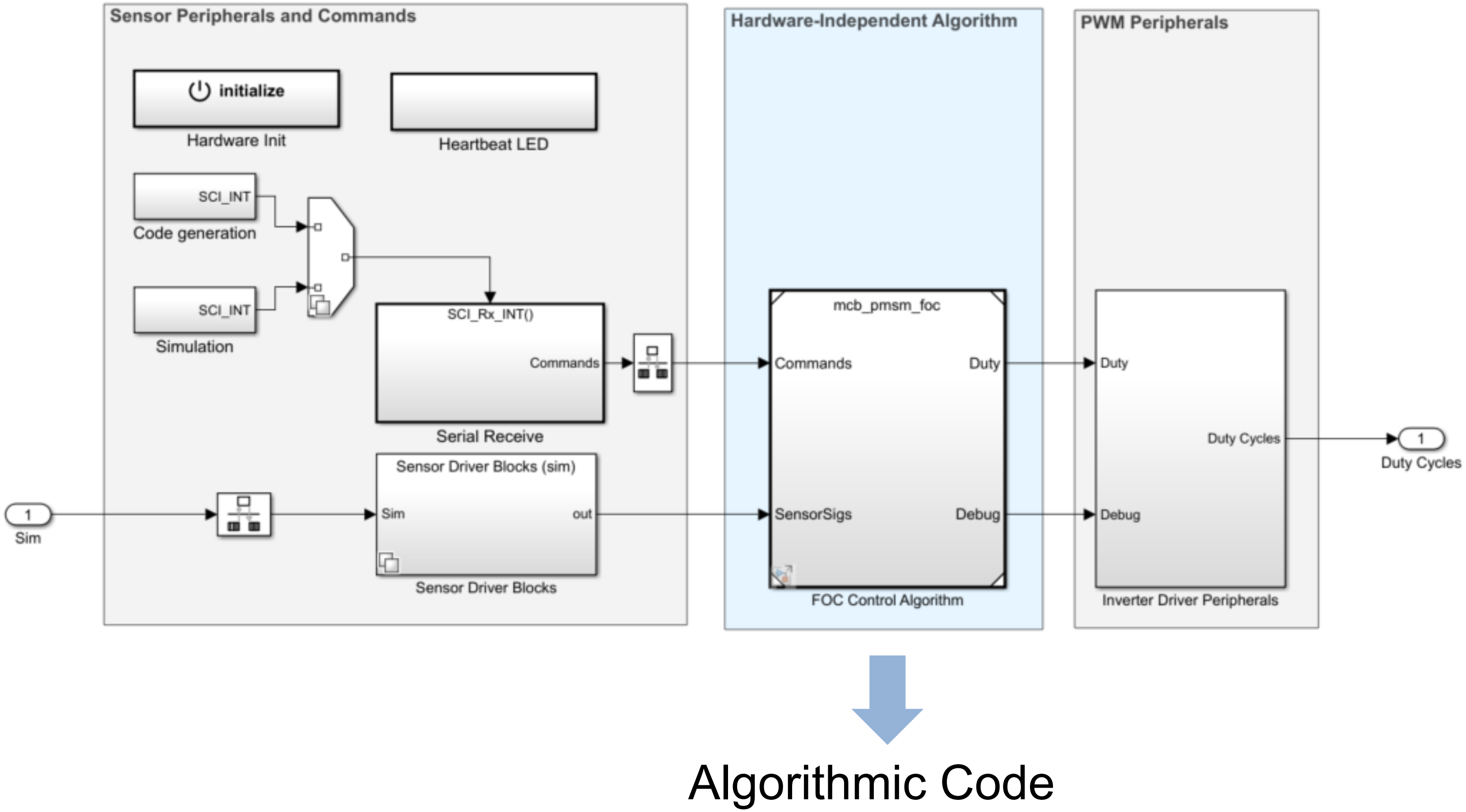
- Model field-oriented control algorithm
- Model sensor decoders or sensorless observers
- Tune loop gains
- Verify in closed-loop simulation



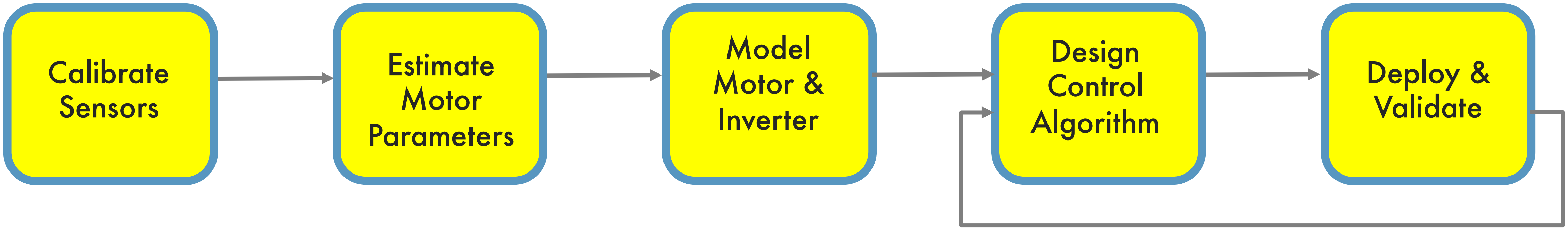
Deployment



- Target NXP S32K MCU with automatically generated C code
- Use provided example to partition the model into algorithmic and hardware-specific parts
- Validate on hardware



Workflow for implementing field-oriented control



DEPLOYING MOTOR CONTROL ALGORITHMS ON NXP S32K FROM SIMULINK

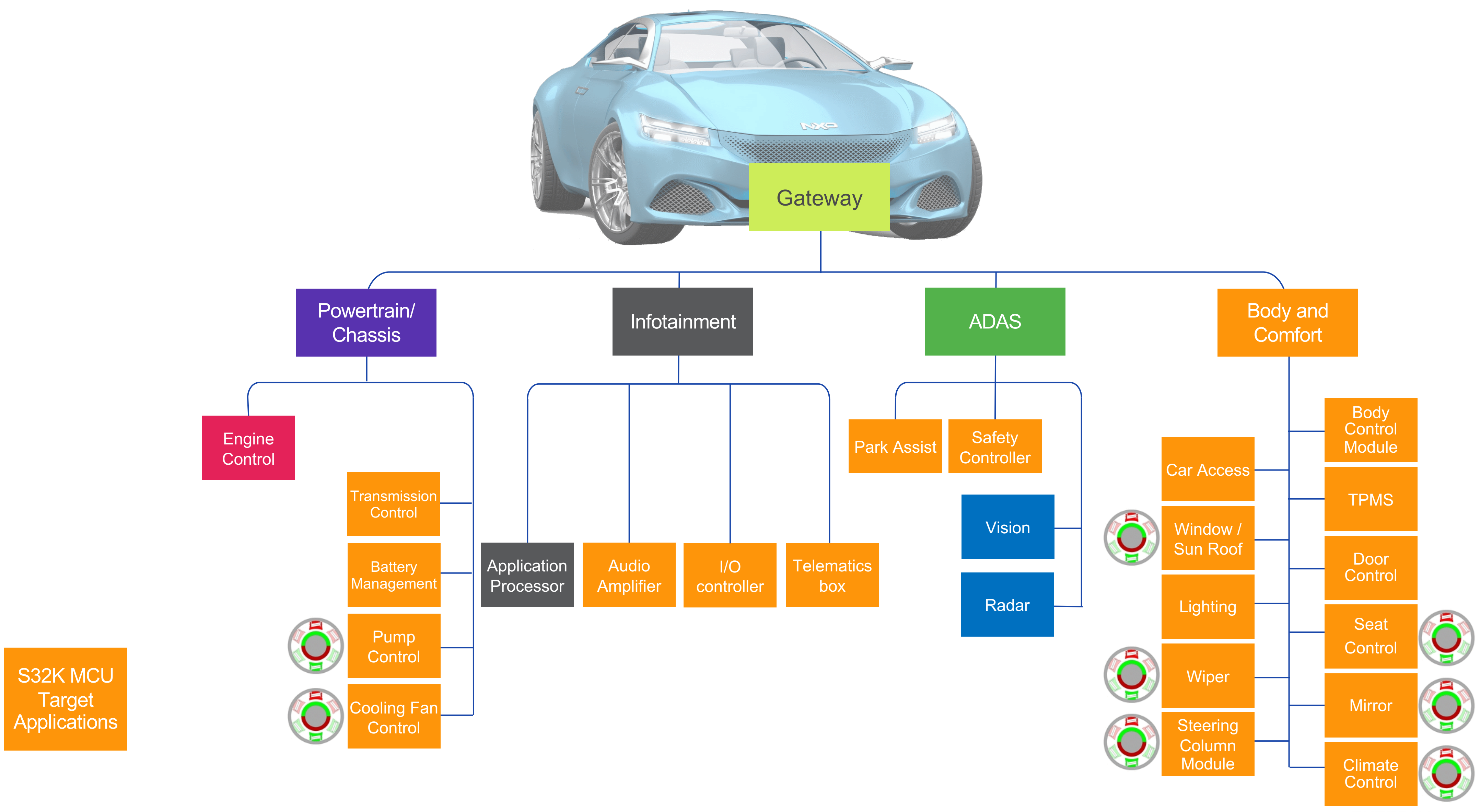
Daniel Popa
MBDT Product Manager & Architect

DECEMBER 2020

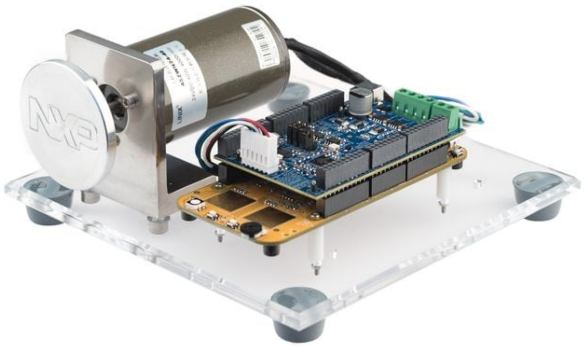
SECURE CONNECTIONS
FOR A SMARTER WORLD

EXTERNAL

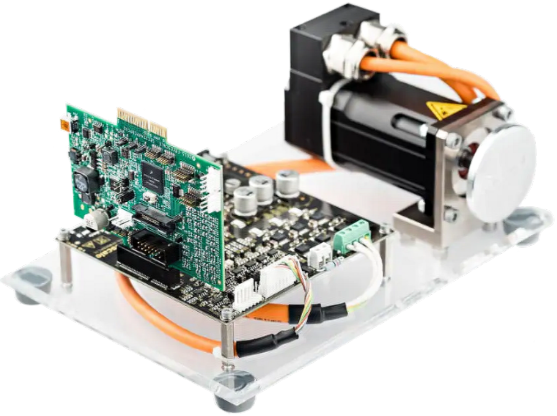
S32K – SOLUTIONS FOR EDGE NODES



NXP MOTOR CONTROL SOLUTIONS



[MCSPTE1AK144]



[MTRCKTSPS5744P]

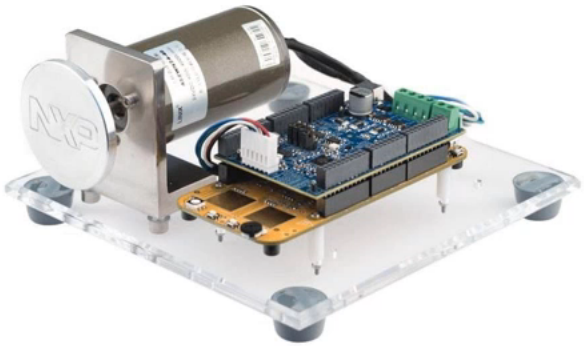
[MTRCKTDBN5643L]

[MCSXTE2BK142]

[MTRCKTSPS5643L]

[MTRCKTDPS5643L]

NXP MOTOR CONTROL SOLUTIONS



MCSPTE1AK144

MTRCKTSPS5744P

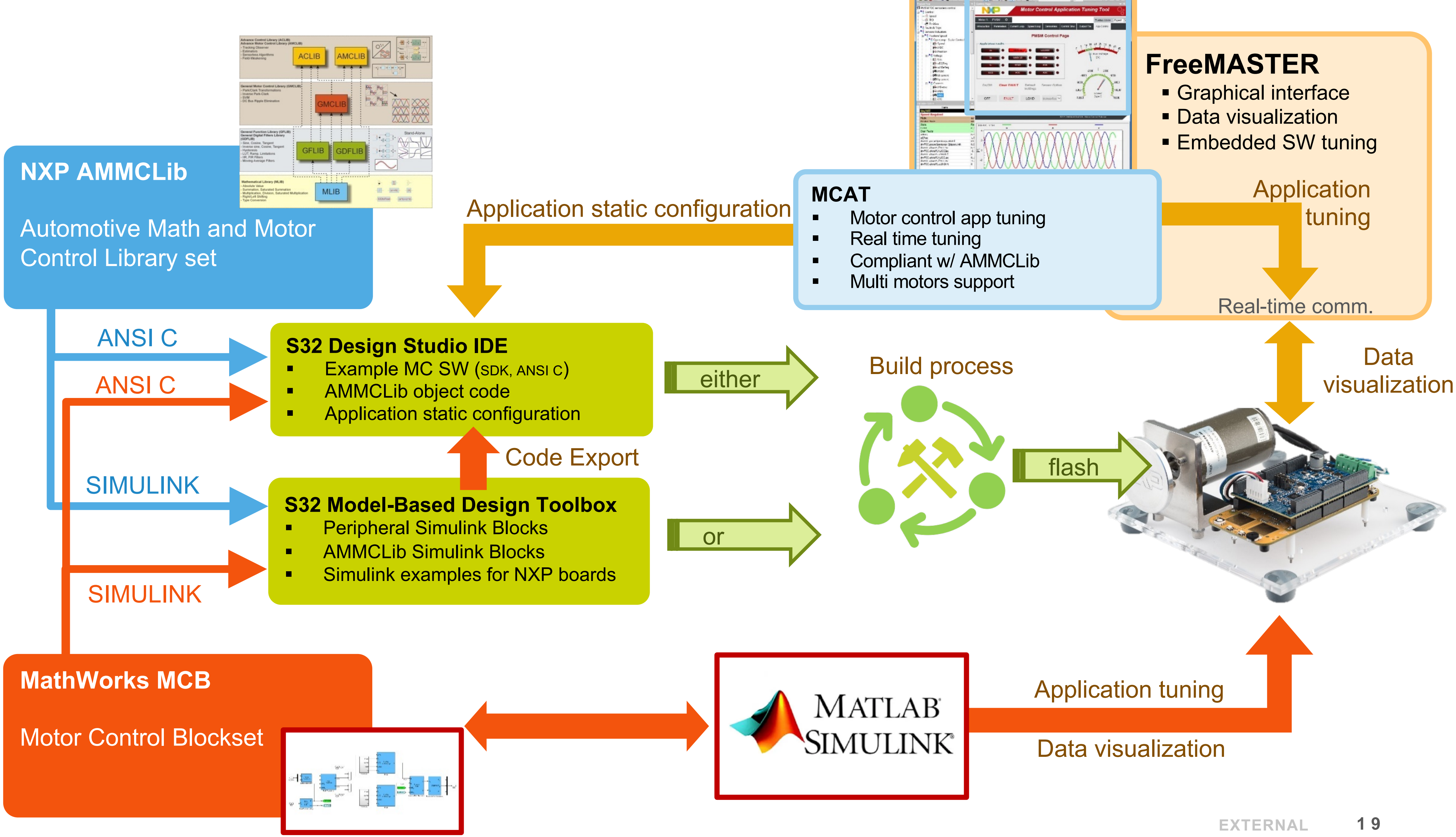
MTRCKTDBN5643L

MCSXTE2BK142

MTRCKTSPS5643L

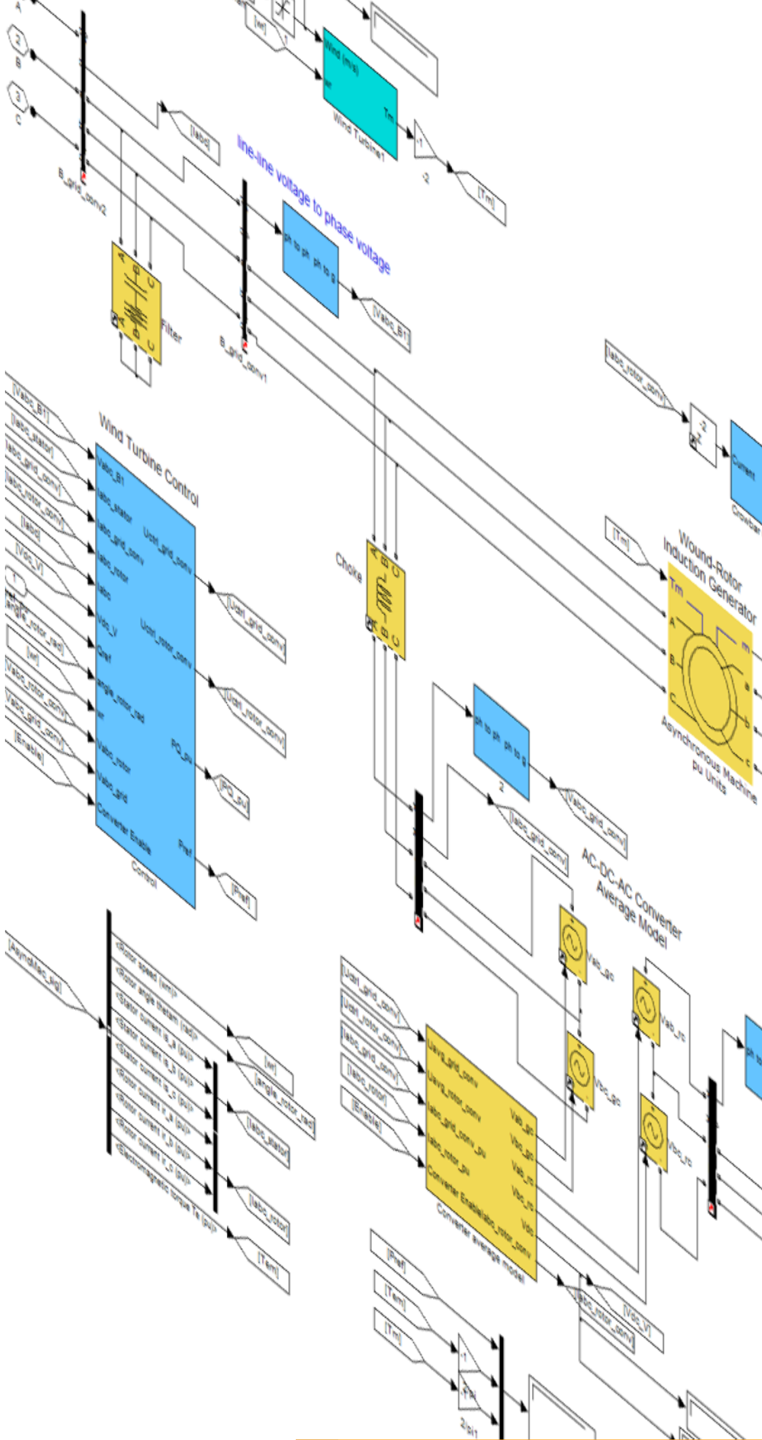
MTRCKTDPS5643L

NXP TOOLCHAIN SW DEVELOPMENT FLOW

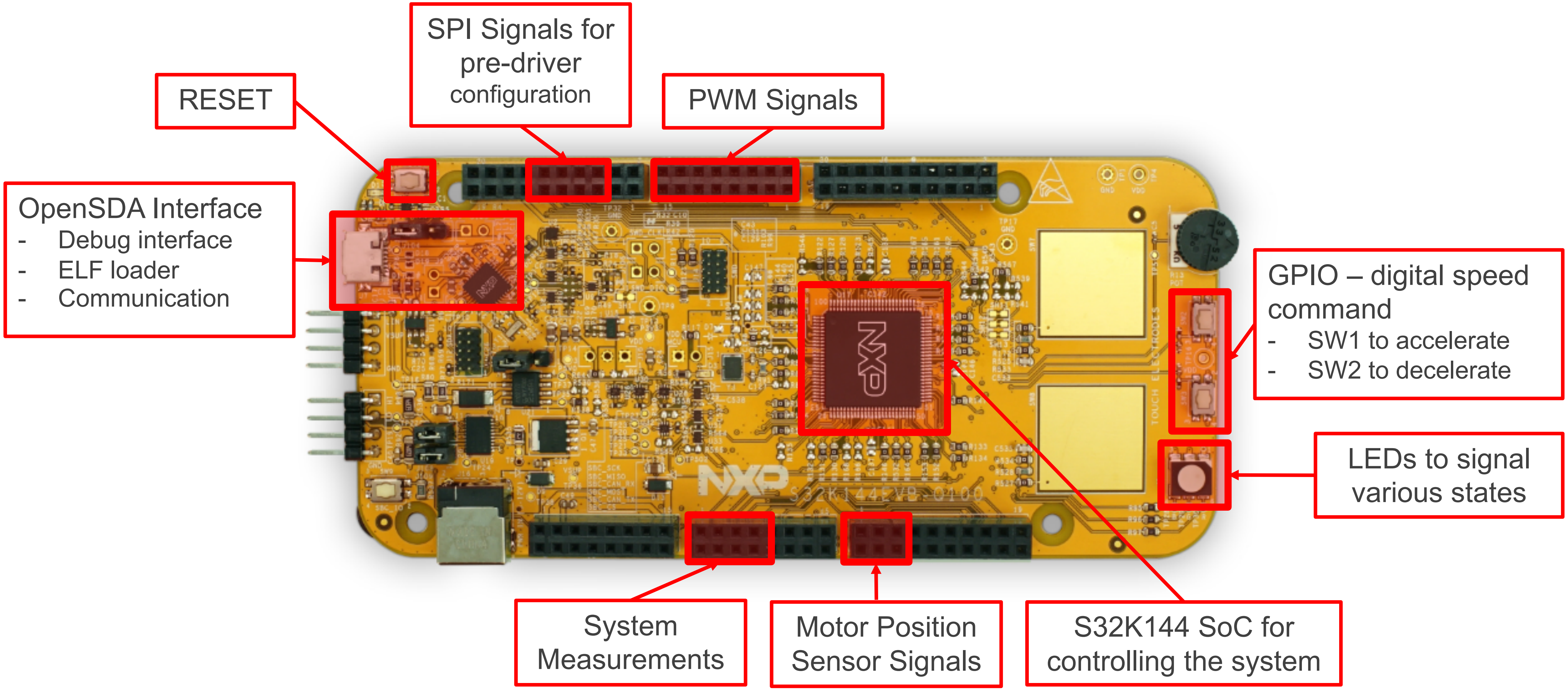


PMSM FOC with MCB on MCSPTE1AK144

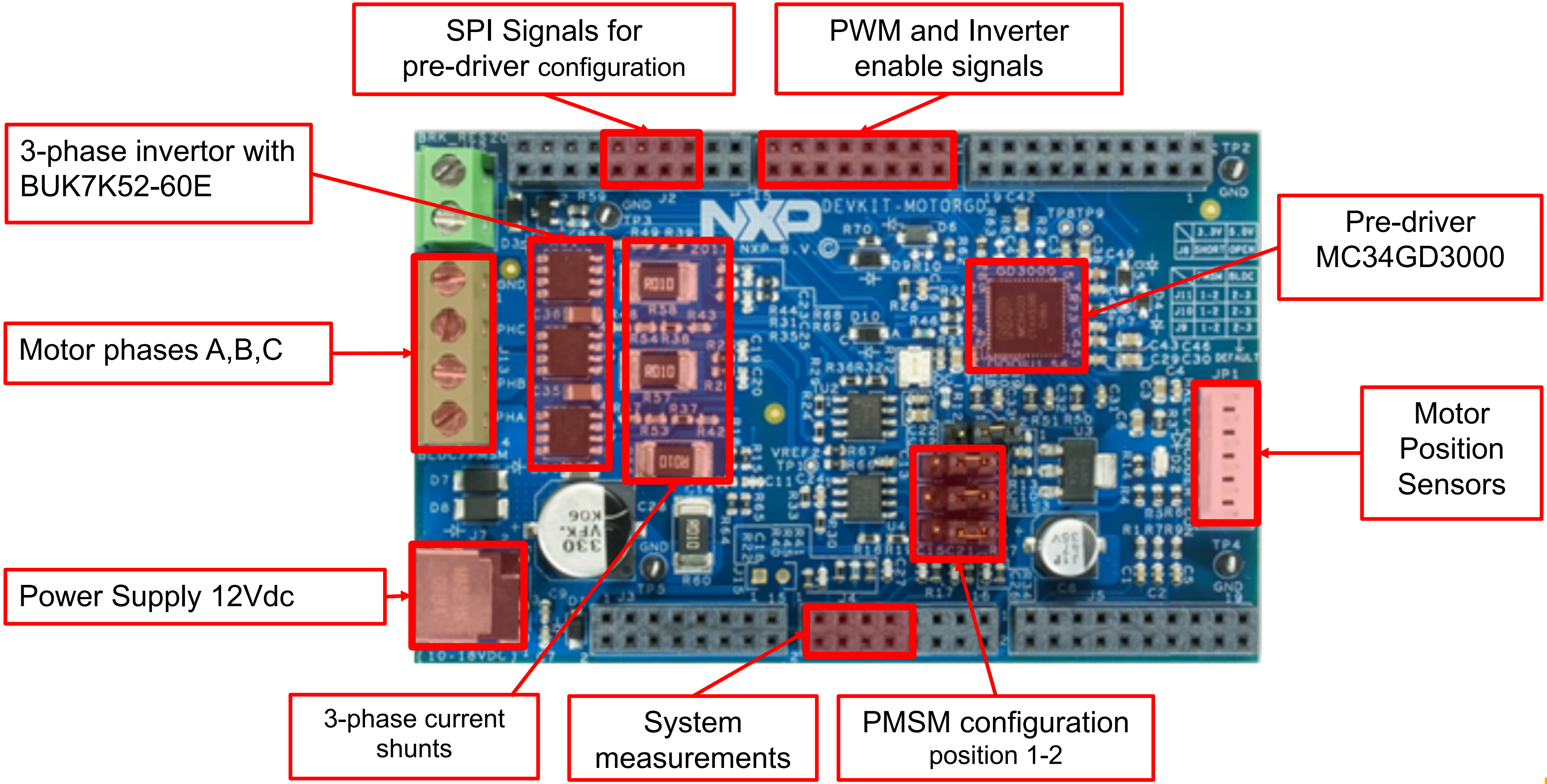
SECURE CONNECTIONS
FOR A SMARTER WORLD



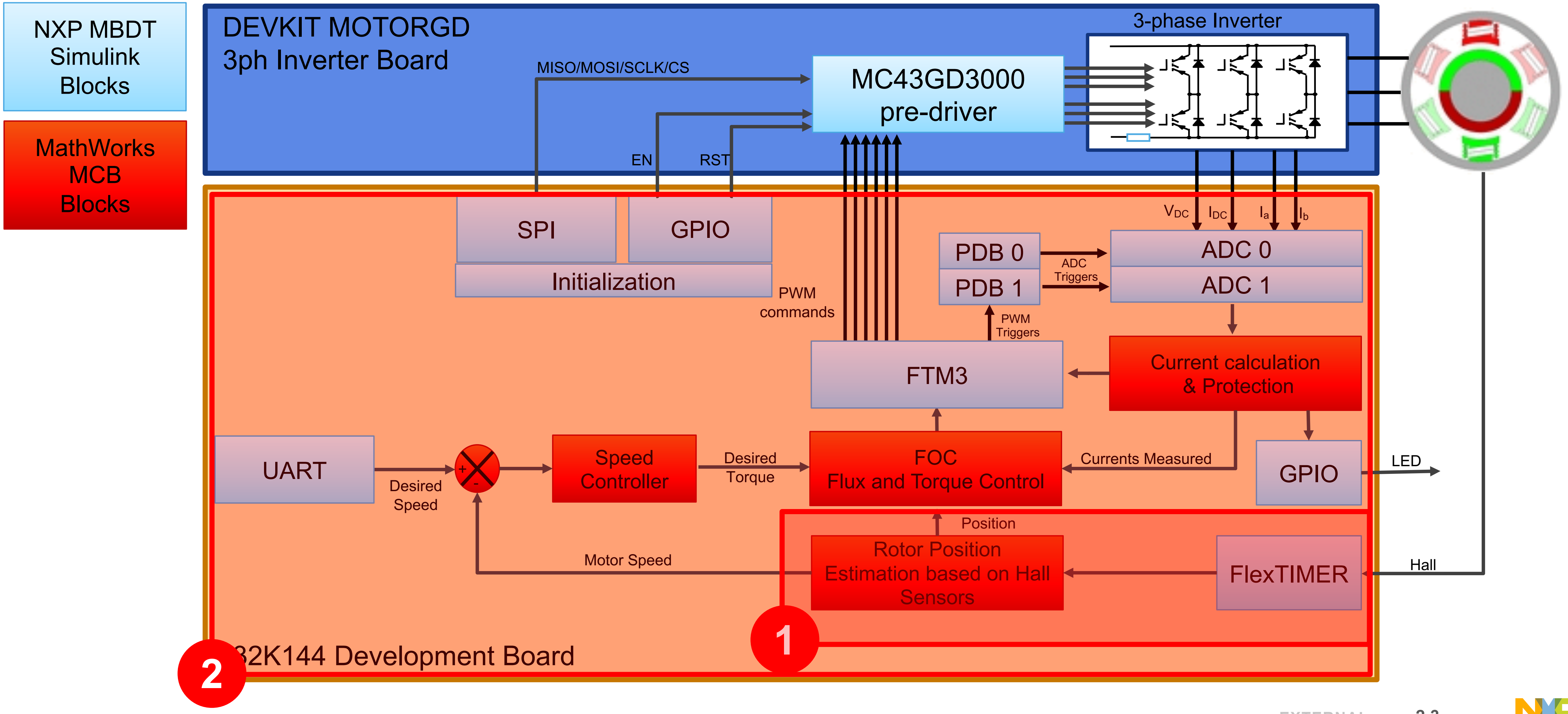
S32K144EVB – PERIPHERALS USED IN THE DEMO

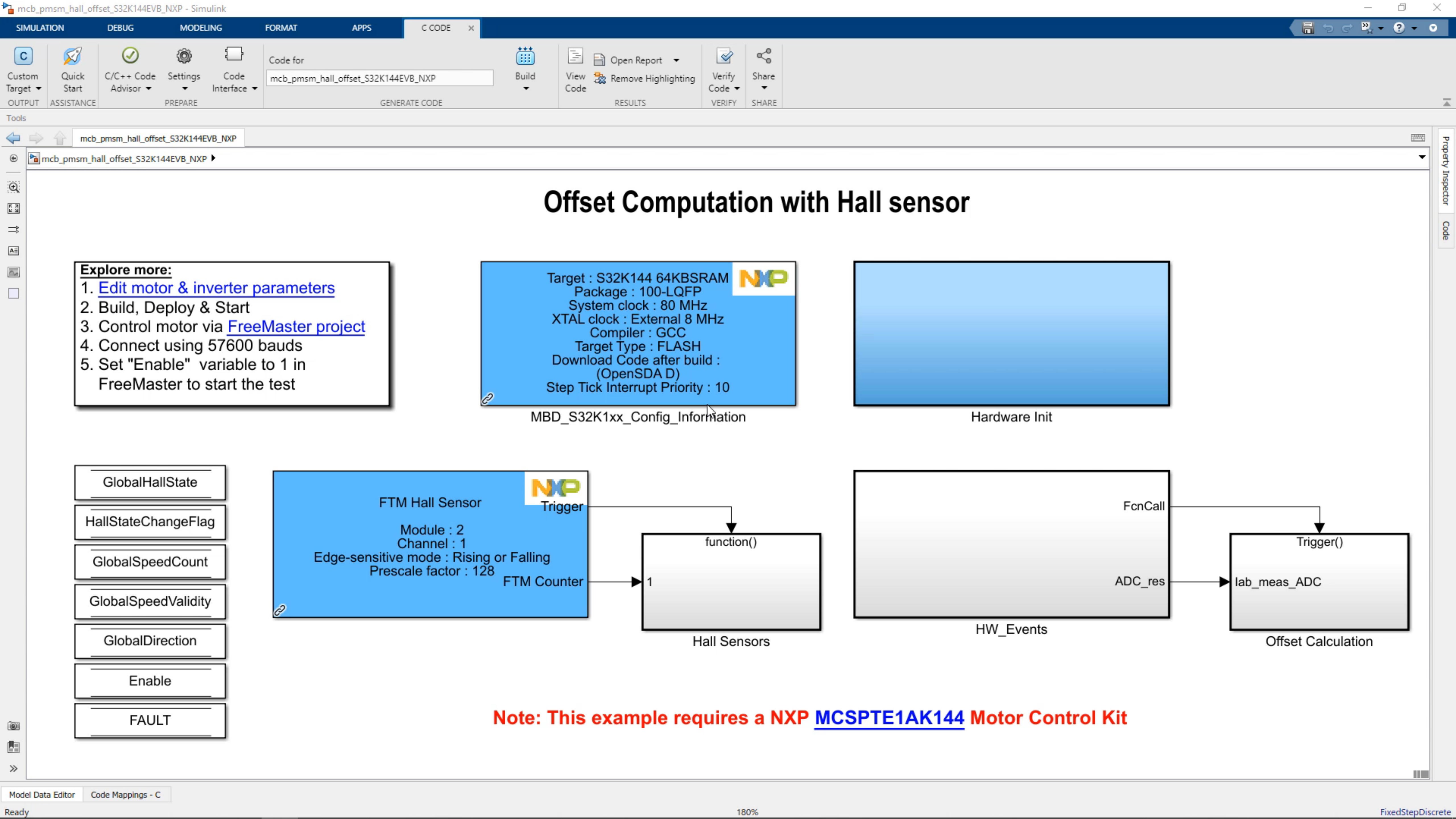


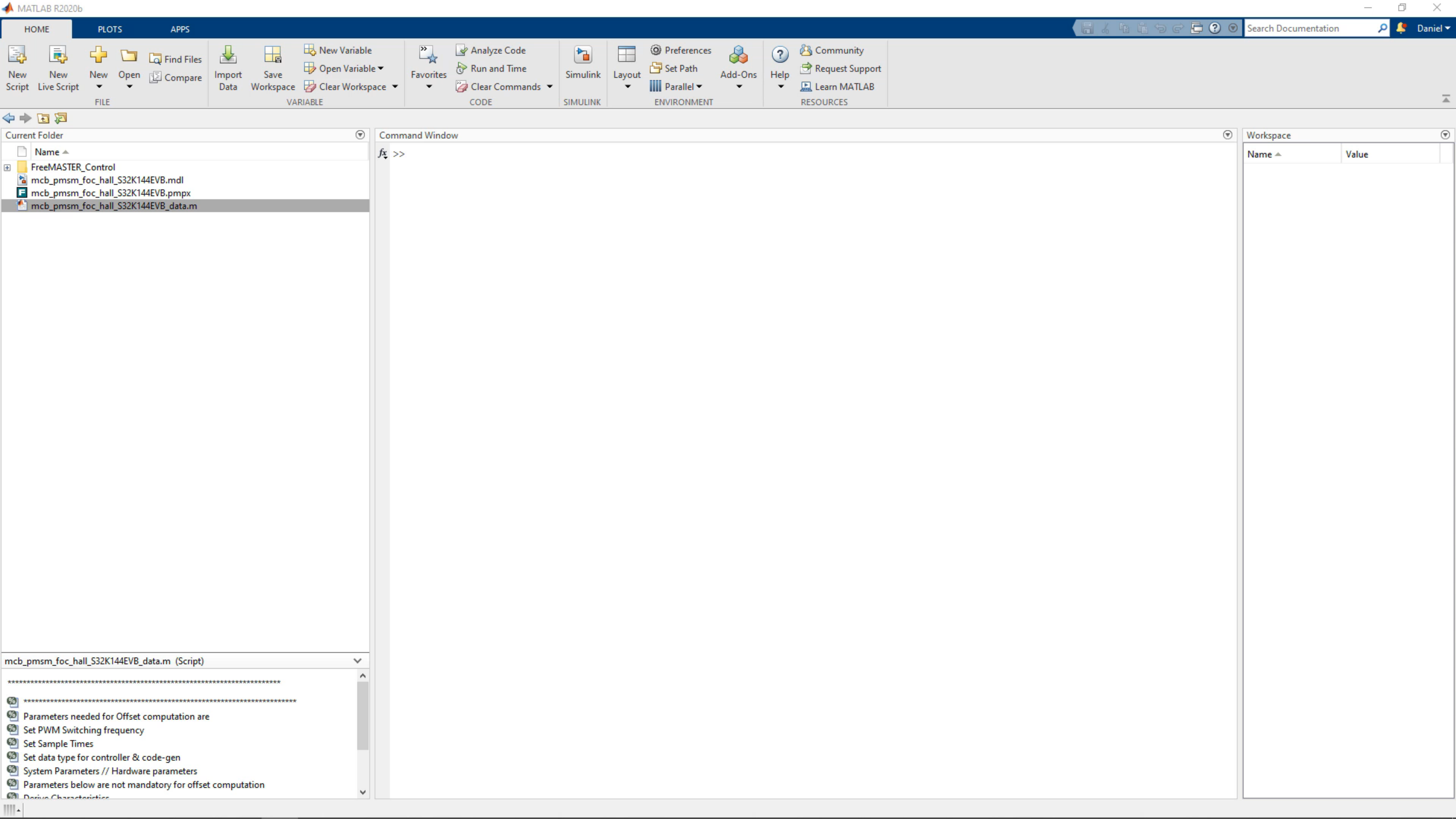
DEVKIT MOTORGD – PERIPHERALS USED IN THE DEMO



PMSM FOC BLOCK DIAGRAM

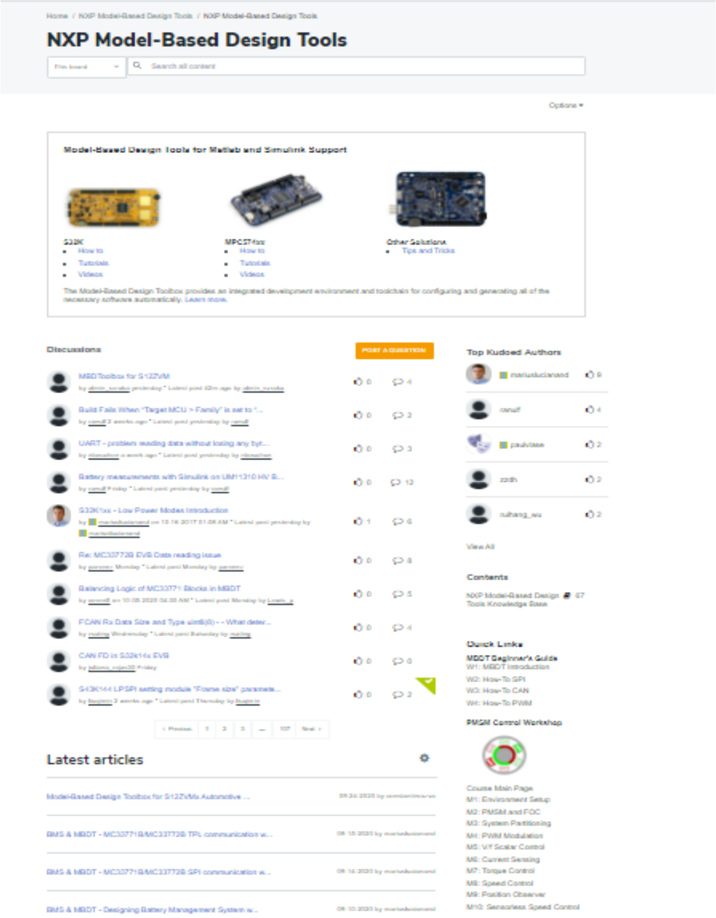



mcb_pmsm_hall_offset_S32K144EVB_NXP - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
C CODE
Custom Target
Quick Start
C/C++ Code Advisor
Settings
Code Interface
Code for
mcb_pmsm_hall_offset_S32K144EVB_NXP
Build
View Code
Open Report
Remove Highlighting
Verify Code
Share
OUTPUT
ASSISTANCE
PREPARE
GENERATE CODE
RESULTS
VERIFY
SHARE
Tools
mcb_pmsm_hall_offset_S32K144EVB_NXP
Offset Computation with Hall sensor
Explore more:
1. Edit motor & inverter parameters
2. Build, Deploy & Start
3. Control motor via FreeMaster project
4. Connect using 57600 bauds
5. Set "Enable" variable to 1 in FreeMaster to start the test
Target : S32K144 64KBSRAM
Package : 100-LQFP
System clock : 80 MHz
XTAL clock : External 8 MHz
Compiler : GCC
Target Type : FLASH
Download Code after build : (OpenSDA D)
Step Tick Interrupt Priority : 10
MBD_S32K1xx_Config_Information
Hardware Init
GlobalHallState
HallStateChangeFlag
GlobalSpeedCount
GlobalSpeedValidity
GlobalDirection
Enable
FAULT
FTM Hall Sensor
Module : 2
Channel : 1
Edge-sensitive mode : Rising or Falling
Prescale factor : 128
Trigger
FTM Counter
function()
1
Hall Sensors
FcnCall
ADC_res
HW_Events
Trigger()
Iab_meas_ADC
Offset Calculation
Note: This example requires a NXP MCSPTE1AK144 Motor Control Kit
Model Data Editor
Code Mappings - C
Ready
180%
FixedStepDiscrete

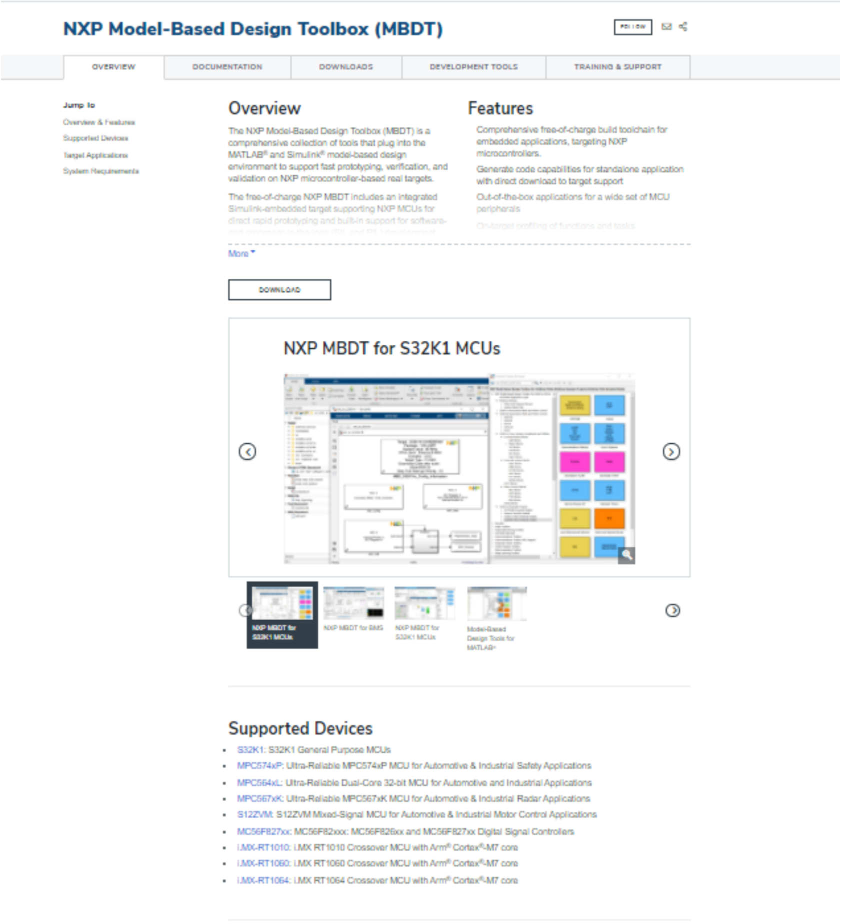

MATLAB R2020b
HOME
PLOTS
APPS
Search Documentation
Daniel
New Script
New Live Script
New
Open
Find Files
Compare
FILE
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
VARIABLE
Favorites
Analyze Code
Run and Time
Clear Commands
CODE
Simulink
SIMULINK
Layout
Preferences
Set Path
Parallel
Add-Ons
ENVIRONMENT
Help
Community
Request Support
Learn MATLAB
RESOURCES
Current Folder
Name
FreeMASTER_Control
mcb_pmsm_foc_hall_S32K144EVB.mdl
mcb_pmsm_foc_hall_S32K144EVB.pmpx
mcb_pmsm_foc_hall_S32K144EVB_data.m
Command Window
>>
Workspace
Name
Value
mcb_pmsm_foc_hall_S32K144EVB_data.m (Script)
Parameters needed for Offset computation are
Set PWM Switching frequency
Set Sample Times
Set data type for controller & code-gen
System Parameters // Hardware parameters
Parameters below are not mandatory for offset computation

GETTING HELP

[MBDT Online Community Examples & Help]



[MBDT home page www.nxp.com/mbdt]



SECURE CONNECTIONS
FOR A SMARTER WORLD